# ホスト*E-LINK*

## 説明書

（株）ファースト

## 御注意

# はしがき

　Ｅ－ＬＩＮＫとは、ＴＣＰ／ＩＰのソケット通信を利用し、ＥｈｔｅｒＮｅｔ上で高速通信を行う通信です。

　この説明書は、以下のバージョンに対応しています。

　　FV904 システム……Ver1.00

　　ホスト E-LINK サンプルライブラリ……Ver1.00

　本書は FV904 に関するターミナルソフト、Winsock ライブラリ、ホスト E-LINK サンプルライブラリについて記載したものです。
　なお、下記の説明書も併せてご参照ください。

　　　◎９０Ｘ操作説明書
　　　◎ＦＶ９０４取扱説明書

# 第１編　E-LINK Terminal 説明書

1 ．E-LINK Terminal について ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・3
2 ．動作環境について ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・3
3 ．インストール方法 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・3
4 ．使用方法 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・4
5 ．画面の説明 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・5
6 ．接続するには ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・6
7 ．接続時のターミナルの操作 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・7
8 ．メニューの説明 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・8
8.1　ファイルメニュー ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・8
8.2　設定メニュー ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・9
9 ．ツリービューの使用方法 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・10
10 ．接続状態の表示について ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・11

# 目　　次

## 第２編　WinSock ライブラリ


1．概要 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 16

1.1　E-LINK ホスト側のプログラム作成について ・・・・・・・・・・・・・・・・・・・・・・・・・・ 17

WinSock の使用を開始(WSAStartup) -------------------------------------------------18
ソケットの作成(socket) -------------------------------------------------------------19
ソケットモード切り替え(ioctlsocket) ------------------------------------------------20
接続(connect) ----------------------------------------------------------------------21
ソケットクローズ(closesocket) -----------------------------------------------------22
データ送信(send) -------------------------------------------------------------------23
データ受信(recv) -------------------------------------------------------------------24
状態取得(select) -------------------------------------------------------------------25
エラー値取得(WSAGetLastError) -----------------------------------------------------27

# 第３編　ホスト E-LINK サンプルライブラリ

E-Link 送信(ELINK_puts) ---------- 33
E-Link 受信(ELINK_gets) ---------- 34
システムコマンド発行(ELINK_outccmd) ---------- 35
ファイルの送信(ELINK_cload) ---------- 36
ファイルの受信(ELINK_cupload) ---------- 37
コマンド応答チェック(ELINK_checkccmdack) ---------- 38
ファイル受信時のヘッダチェック(ELINK_headercheck) ---------- 39
**ファイル転送方法 ・・・・・・・・・・ 40**
**通信速度 ・・・・・・・・・・ 42**

# 第1編

# E-LINK Terminal説明書

# 1. E-LINK Terminal について

Ｅ－ＬＩＩＮＫ ＴｅｒｍｉｎａｌはＦＶ９０４にオプションとして搭載されている
Ｅ－ＬＩＮＫユニットを通じて、ＦＶ９０４をコントロールするツールです。

# 2. 動作環境について

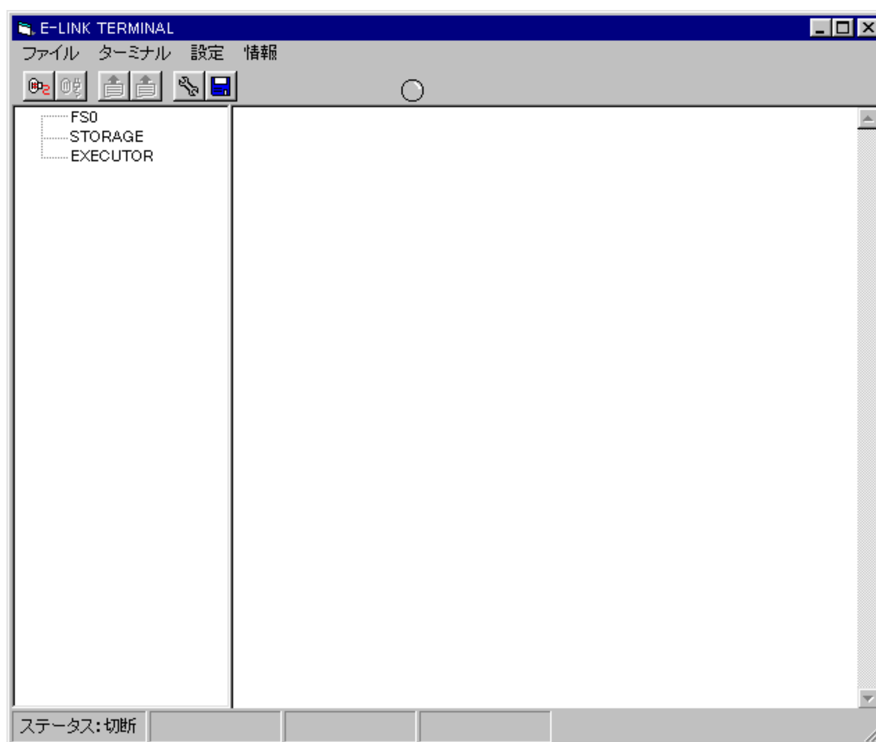
○ Windows95,98,NT が動作するＰＣ
○ ＴＣＰ／ＩＰが使用できるネットワークカードが搭載されていること
　ＴＣＰ／ＩＰプロトコルがインストールされ動作していること
　（Windows95,98,NT は Microsoft 社の製品です）

# 3. インストール方法

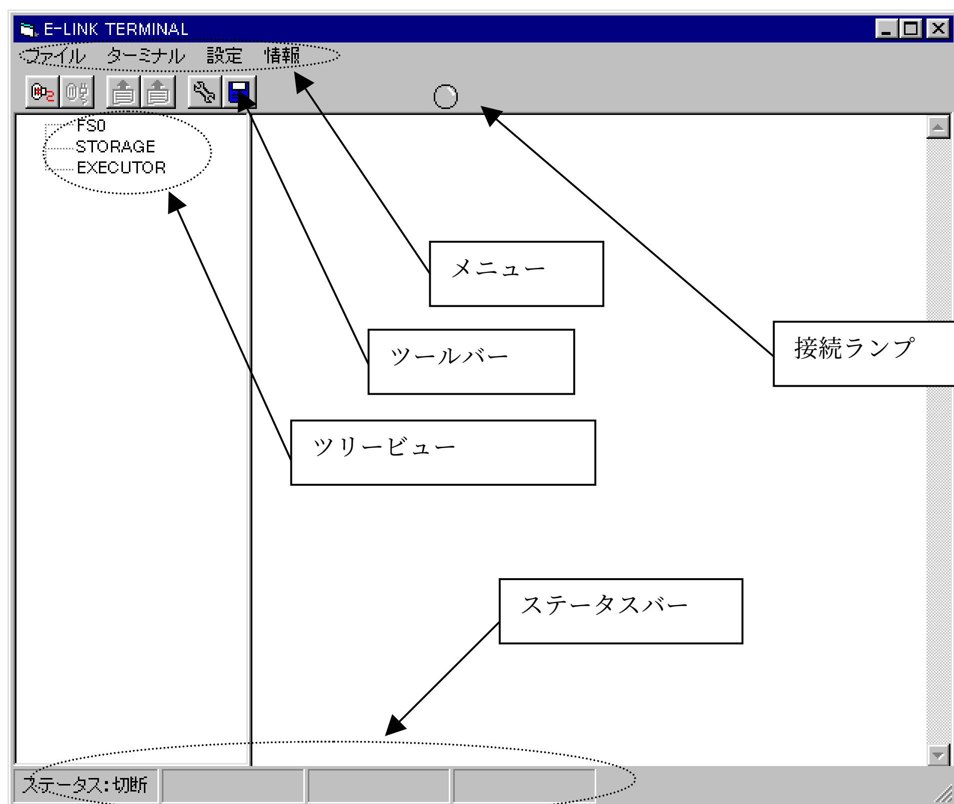
Setup.exe を起動して指示に従ってください。

# 4．使用方法

実行ファイルを実行すると以下のような画面が表示されます。



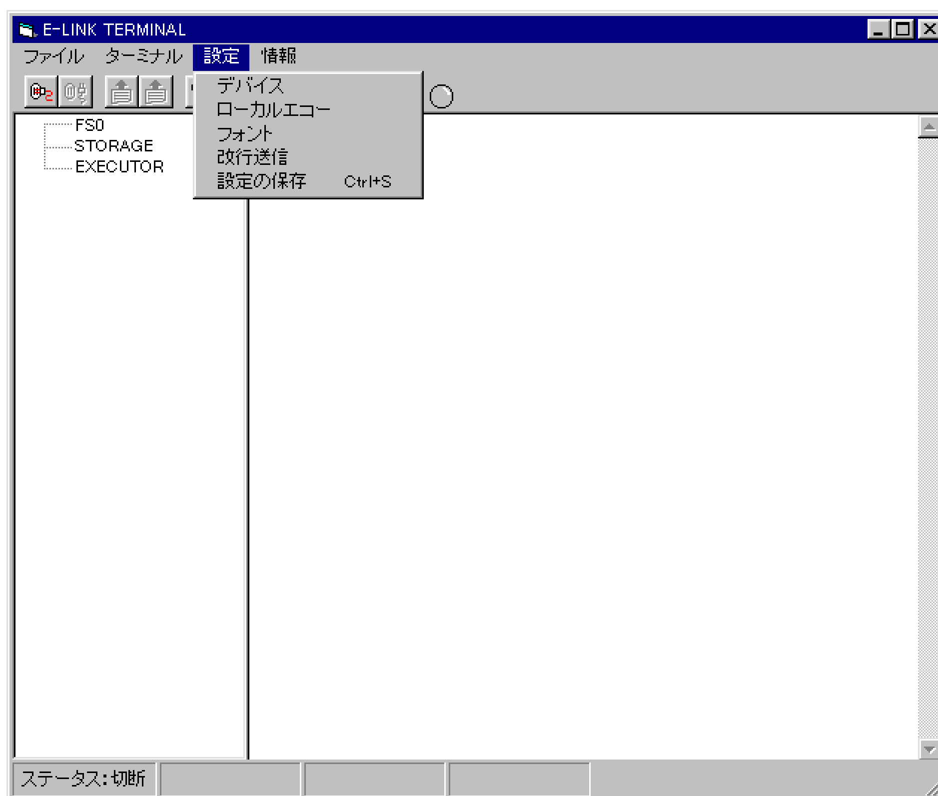
# 5．画面の説明



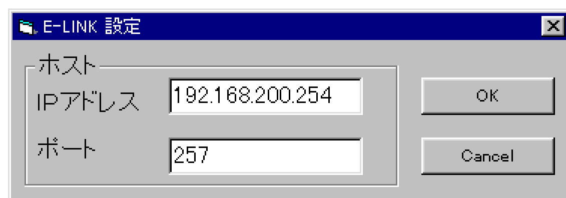

以降、上記の名称で説明します。

# 6．接続するには

ターミナルソフトとして使用するには、設定→デバイスを選択し、

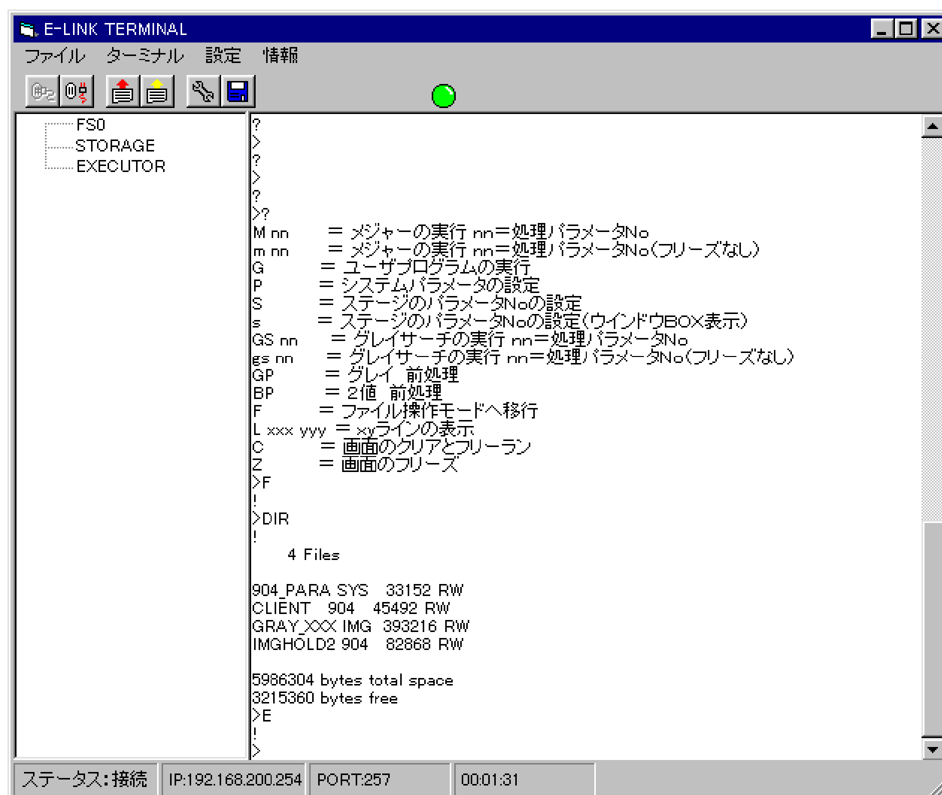



IP アドレス、ポートを９０４の E-LINK の設定にあわせてください。
その後、ターミナル→接続で接続可能です。




**接続する９０４の設定は システム→入出力→オプション→標準 I/O を E_LINK とし、エコーバックをＯＮ、ターミネートを CR/LF として下さい。**

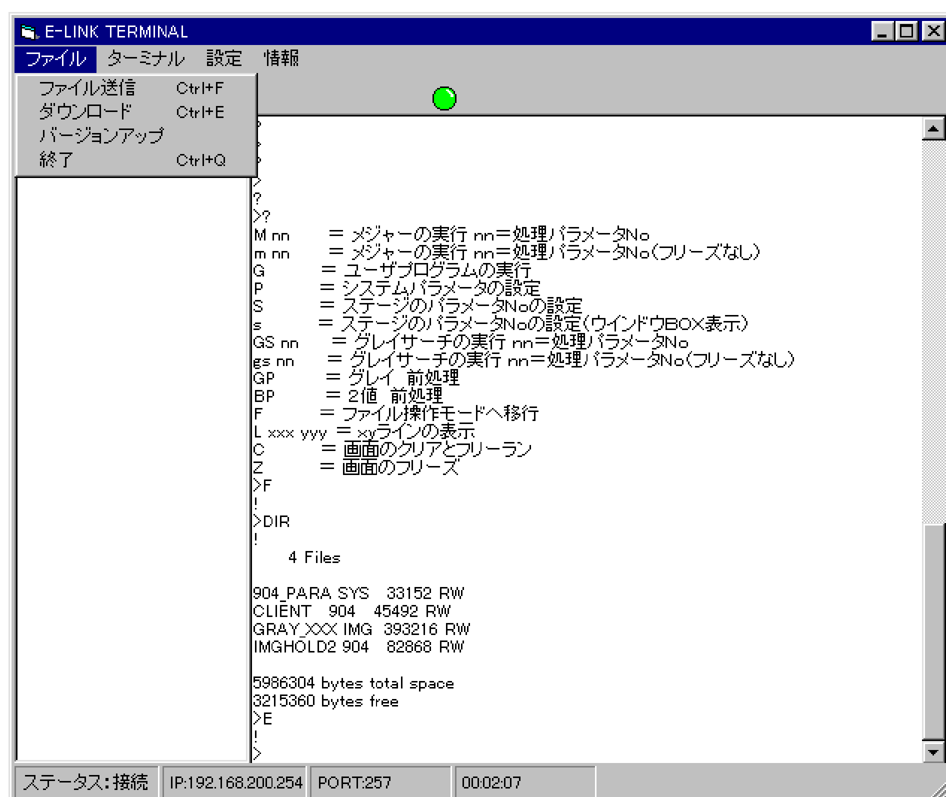
# 7．接続時のターミナルの操作




接続後は、ターミナルの右画面に入力できます。
切断時は、入力はできません。
上の図は、システムコマンドを実行した例です。

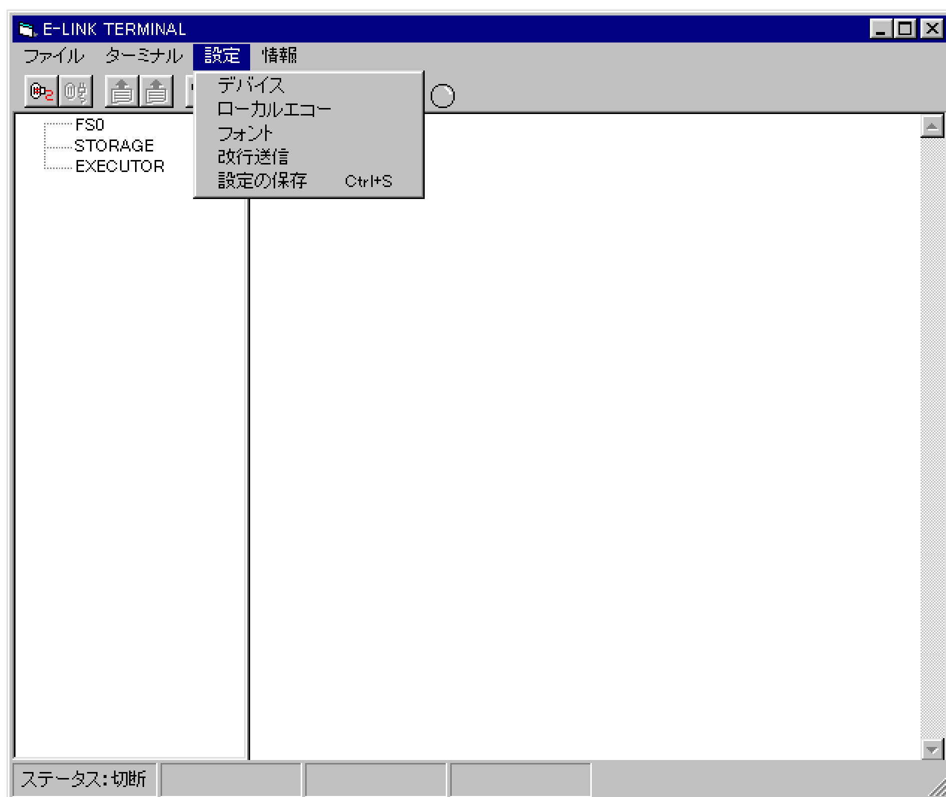
# 8．メニューの説明

## 8.1　ファイルメニュー




ファイル送信 ････････････接続先の９０４のＦＳ０にファイルを送信します。
ダウンロード ････････････接続先の９０４の EXECUTOR にファイルを送信します。
バージョンアップ ････････接続先の９０４のシステムファイルをバージョンアップします。
　　　　　　　　　　　　　実行時は９０４のシステムが保存されているディレクトリを
　　　　　　　　　　　　　指定してください。
終了 ････････････････････本プログラムを終了します。

# 8.2　設定メニュー




デバイス ・・・・・・・・・・・・・・・ Ｅ－ＬＩＮＫのＩＰアドレスなどを設定します。
ローカルエコー ・・・・・・・・・・ ローカルエコーをするか設定します。
フォント ・・・・・・・・・・・・・・・ ツリービュー、ターミナル画面のフォントを設定します。
改行送信 ・・・・・・・・・・・・・・・改行の入力によって入力された文字を送信します。
設定の保存 ・・・・・・・・・・・・・ 上記設定を保存します。

# 9．ツリービューの使用方法

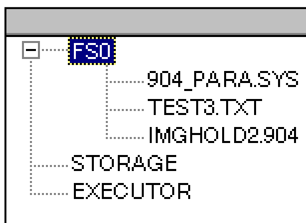
起動時には９０４のディレクトリが表示されています。



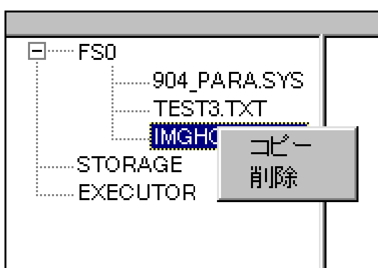

ツリービューは以下のような使用方法があります。




接続状態で、ディレクトリをクリックすると ＋ が表示されます。
これをクリックすると、




ＦＳ０内のファイルが表示されます。
この作業は、STORAGE、EXECUTOR も同様です。
さらに表示されたファイル上でをマウスの右ボタンでクリックすると、
以下のようなメニューが表示され、「ＰＣにコピー」もしくは「削除」ができます。

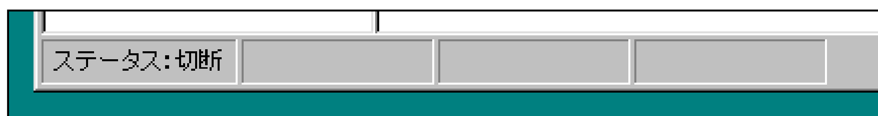



また、接続された状態で**<u>エクスプローラー等からファイルをディレクトリ上にドラッグ＆ドロップすることも可能</u>**です。

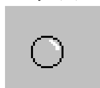
# 10．接続状態の表示について

接続の状態は、以下のような機能によって確認できます。

### 切断時：




**ステータスバーの表示**

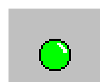


**接続ランプの表示**

### 接続時：




**左から、現在の状態、接続先ＩＰアドレス、ポート、接続時間となります。**



**灰色から緑色に変化します。**

# 第２編

# WinSockライブラリ

# アルファベット順ライブラリ目次


| 項番 | ライブラリ名 | ページ |
| --- | --- | --- |
| 1 | Closesocket | 22 |
| 2 | Connect | 21 |
| 3 | Ioctlsocket | 20 |
| 4 | Recv | 24 |
| 5 | Select | 25 |
| 6 | Send | 23 |
| 7 | Socket | 19 |
| 8 | WSAGetLastError | 27 |
| 9 | WSAStartup | 18 |

# ■WinSockライブラリ

#include <winsock.h> もしくは #include <winsock2.h>

### WinSock の使用を開始

```
int WSAAPI WSAStartup( WORD wVersionRequested, LPWSADATA lpWSAData );
```

### ソケットの作成

```
socket( int af, int type, int protocol );
```

### ソケットモード切り替え

```
int ioctlsocket( SOCKET s, long cmd, u_long FAR* argp );
```

### 接続

```
int connect( SOCKET s, const struct sockaddr FAR* name, int namelen );
```

### ソケットクローズ

```
int closesocket( SOCKET s );
```

### データ送信

```
int send ( SOCKET s, const char FAR * buf, int len, int flags );
```

### データ受信

```
int recv ( SOCKET s, const char FAR * buf, int len, int flags );
```

### 状態取得

```
int select ( int nfds, fd_set FAR * readfds, fd_set FAR * writefds,
             fd_set FAR * exceptfds, const struct timeval FAR * timeout );
```

### エラー値取得

```
int WSAGetLastError( void );
```

# １．概要

E-LINK とは TCP ソケットプログラムを利用した高速通信です。
E-LINK ユニットの搭載によって、９０４はイーサネット上で、通信を行う事が可能です。
また、ホストＰＣなどが TCP ソケット(WinSock など)の使用できる環境であれば、
ソケットプログラムを作成する事により、９０４との通信が可能となります。

## 動作環境

○Windows 95、98、NT で、TCP/IP が使用できるもの
○イーサネットボード

# 1.1 E-LINK ホスト側のプログラム作成について

E-LINK は TCP ソケットプログラムを利用して通信が可能です。
ホスト側（ＰＣなど）は TCP ソケット（WinSock など）で作成できます。
WinSock は現在の最新版は Winsock2.0 です。
Windows 95 では Winsock1.1 がサポートされています。
Windows NT,98 では Winsock2.0 がサポートされています。

**なお、弊社では WinSock ライブラリに関するサポートは行っておりません。**
**WinSock の詳しい資料は MSDN(MicroSoftDeveloperNetwork)などに掲載されております。**

以下に Winsock1.1 の代表的なライブラリを紹介します。

## 参考文献 及び 参考Ｗｅｂ


ＴＣＰ／ＩＰハンドブック
若林 宏 著
秀和システム
ISBN4-87966-850-8

ＷｉｎＳｏｃｋ２．０プログラミング
Lewis Napper 著
ソフトバンク社
ISBN4-7973-0688-2

PC-TCPIP-FAQ-J
http://www.naoe.hiroshima-u.ac.jp/staffs/hirata/pc-tcpip/

The WinSock channel
http://www.stardust.com/winsock/index.htm

# *WSAStartup*

**機　能**

WinSock の使用を開始

**形　式**

```
#include <winsock.h> もしくは #include <winsock2.h>
int WSAAPI WSAStartup( WORD wVersionRequested, LPWSADATA lpWSAData );
```

**解　説**

WinSock の使用を開始します。

① ***wVersionRequested*** はプログラムで使用する Winsock のバージョンを指定します。
　　最上位バイトがメジャーバージョン、
　　最下位バイトがマイナーバージョンとなります。

② ***lpWSAData*** は WSADATA 構造体へのポインタです。

**戻り値**

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| 0 | ありません | 正常終了 |
| エラーコード | | |

**留意事項**

○エラー発生時、WSAGetLastError() を呼び出してはいけません。

○wVersionRequested には、MAKEWORD マクロ を使用すると便利です。

○Winsock1.1 の場合は、#include <winsock.h>をインクルードしてください。
　Winsock2.0 の場合は 、#include <winsock.h>もしくは#include <winsock2.h>をインクルードしてください(以下同一)

# socket

## 機　能

ソケットの作成

## 形　式

```
#include <winsock.h> もしくは #include <winsock2.h>
socket( int af, int type, int protocol );
```

## 解　説

ソケット記述子の作成

① ***af*** はアドレスファミリを指定します。

| 値 | 定　数 | 意　味 |
|---|---|---|
| 2 | AF_INET | TCPの場合 |

② ***type*** はソケットタイプを指定します。

| 値 | 定　数 | 意　味 |
|---|---|---|
| 1 | SOCK_STREAM | TCPの場合 |

③ ***protocol*** はプロトコルを指定します。

| 値 | 定　数 | 意　味 |
|---|---|---|
| 6 | IPPROTO_TCP | TCPの場合 |

## 戻り値

| 値 | 定　数 | 意　味 |
|---|---|---|
| 1～ | ありません | ソケット記述子 |
| 0xFFFF | INVALID_SOCKET | WSAGetLastError()で確認してください |

## 留意事項

ありません。

# *ioctlsocket*

**機　能**　　ソケットモード切り替え

**形　式**

```
#include <winsock.h> もしくは #include <winsock2.h>
int ioctlsocket( SOCKET s, long cmd, u_long FAR* argp );
```

**解　説**　　ソケットのＩ／Ｏモードの切り替え

ブロッキングソケットから非ブロッキングソケットに切り替えます。またはその逆。

① ***s*** はソケット記述子です。

② ***cmd*** はコマンドです。

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| ありません | FIONBIO | TCPの場合 |

③ ****argp*** は cmd に関する引数のポインタです。

**戻り値**

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| ０ | ありません | 正常終了 |
| －１ | SOCKET_ERROR | WSAGetLastError()で確認してください |

**留意事項**　ありません。

# connect

## 機　能

接続

## 形　式

```
#include <winsock.h> もしくは #include <winsock2.h>
int connect( SOCKET s, const struct sockaddr FAR* name, int namelen );
```

## 解　説

指定されたソケットへの接続

① ***s*** はソケット記述子です。

② ****name*** は接続先の情報が入力された sockaddr 構造体のポインタです。

③ ***namelen*** は sockaddr 構造体の大きさです。

## 戻り値

| 値 | 定　数 | 意　味 |
|---|---|---|
| 0 | ありません | 正常終了 |
| －1 | SOCKET_ERROR | WSAGetLastError()で確認してください |

## 留意事項

ありません。

# closesocket

**機　能**　ソケットクローズ

**形　式**

```
#include <winsock.h> もしくは #include <winsock2.h>
int closesocket( SOCKET s );
```

**解　説**　ソケットをクローズします。

① ***s*** はソケット記述子です。

**戻り値**

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| ０ | ありません | 正常終了 |
| －１ | SOCKET_ERROR | WSAGetLastError()で確認してください |

**留意事項**　ありません。

# send

## 機　能

データ送信

## 形　式

```
#include <winsock.h> もしくは #include <winsock2.h>
int send ( SOCKET s, const char FAR * buf, int len, int flags );
```

## 解　説

データを送信します。

① ***s*** はソケット記述子です。

② ****buf*** は送信データのバッファへのポインタです。

③ ***len*** は送信データの長さです。

④ ***flags*** はオプション、または０です。

## 戻り値

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| １～ | ありません | 送信データ量 |
| －１ | SOCKET_ERROR | WSAGetLastError()で確認してください |

## 留意事項

ありません。

# recv

## 機　能

データ受信

## 形　式

```
#include <winsock.h> もしくは #include <winsock2.h>
int recv ( SOCKET s, const char FAR * buf, int len, int flags );
```

## 解　説

データを受信します。

① ***s*** はソケット記述子です。

② ****buf*** は受信データのバッファへのポインタです。

③ ***len*** は受信データの長さです。

④ ***flags*** はオプション、または０です。

## 戻り値

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| ０ | ありません | 接続がクローズされた |
| １～ | ありません | 受信データ量 |
| －１ | SOCKET_ERROR | WSAGetLastError()で確認してください |

## 留意事項

ありません。

# select

## 機　能

状態取得

## 形　式

```
#include <winsock.h> もしくは #include <winsock2.h>
int select ( int nfds, fd_set FAR * readfds, fd_set FAR * writefds,
                fd_set FAR * exceptfds, const struct timeval FAR * timeout );
```

## 解　説

ソケットの現在の状態を調べます。

① ***nfds*** はＢＳＤソケットと互換を持たすためのダミー変数です。

② ****readfds*** は読み取り可能かをチェックする fd_set 構造体のポインタです。

③ ****writefds*** は書き込み可能かをチェックする fd_set 構造体のポインタです。

④ ****exceptfds*** はエラーをチェックする fd_set 構造体のポインタです。

⑤ ***timeout*** は最大ブロッキング時間を定義します。NULL の場合は、データが受信されるまでブロッキングされます。

## 戻り値

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| ０ | ありません | タイムアウト |
| １～ | ありません | 指定の条件を満たすソケット記述子の総数 |
| －１ | SOCKET_ERROR | WSAGetLastError()で確認してください |

# 留意事項

次の環境で動作する Microsoft(R) Win32(R) Application Programming Interface (API)
Microsoft(R) Windows(R) 95 OEM Service Release Version 2
Microsoft(R) Windows(R) 95

Winsock の select() API は、ノンブロッキング ソケット上でブロックに失敗することがあります。
その後で send() および recv() が呼び出されると、エラーとして WSAEWOULDBLOCK が返されます。
select() は読み取るデータが存在していることを示し正常にリターンしますが、
recv() 呼び出しは、現在有効なデータが存在していないことを示すエラー コード
WSAEWOULDBLOCK でリターンします。この現象は、Windows NT 4.0 では発生しません。

## 解決方法

この問題に対処するには、次のいずれかの方法を行ってください。

○一番シンプルな対処方法は、WSAEWOULDBLOCK エラー コードを明示的に無視し、一呼吸おいて recv() をもう一度呼び出す方法（ポーリング）です。

○select() の代わりに WSAAsyncSelect() を使用します。
WSAAsyncSelect() を使用すると、ソケットに関連付けられた Windows メッセージを送信することにより、ソケットの活動状況の非同期通知を実現することができます。

○select() の代わりに WSAEventSelect() を使用します。
（ただしこの方法は、 Winsock2 for Windows 95 がインストールされている場合にのみ有効です。）
WSAEventSelect() を使用すると、ソケットに関連付けられたイベントにシグナルを送ることで、ソケットの活動状況の非同期通知を実現することができます。
（以上 マイクロソフト社 ＭＳＤＮ 文書番号: J041110 より抜粋）

# *WSAGetLastError*

**機　能**　　エラー値取得

**形　式**　　#include <winsock.h> もしくは #include <winsock2.h>

```
int WSAGetLastError( void );
```

**解　説**　　最後におきたエラーを取得します。

## 戻り値

エラーとなった最後の操作のエラーコード。エラー値一覧を参照してください

### エラー値 一覧

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| 10004 | WSAEINTR | 関数コールが中止されました |
| 10013 | WSAEACCES | アクセス権限を持っていません |
| 10014 | WSAEFAULT | 不正アドレス |
| 10022 | WSAEINVAL | 引数が無効です |
| 10024 | WSAEMFILE | ソケット数が上限です |
| 10035 | WSAEWOULDBLOCK | 操作はブロッキングされます |
| 10036 | WSAEINPROGRESS | ブロッキング中です |
| 10037 | WSAEALREADY | 既にプロセスが実行中です |
| 10038 | WSAENOTSOCK | 無効なソケットに操作を行っています |
| 10039 | WSAEDESTADDRREQ | 接続先のアドレスが必要です |
| 10041 | WSAEPROTOTYPE | プロトコルのタイプが不正です |
| 10042 | WSAENOPROTOOPT | プロトコルオプションが不正です |
| 10043 | WSAEPROTONOSUPPORT | サポート外のプロトコルです |
| 10044 | WSAESOCKTNOSUPPORT | サポート外のソケットです |
| 10045 | WSAEOPNOTSUPP | サポート外のオプションです |
| 10046 | WSAEPFNOSUPPORT | サポート外のプロトコルファミリです |
| 10047 | WSAEAFNOSUPPORT | サポート外のアドレスファミリです |
| 10048 | WSAEADDRINUSE | アドレスが使用済みです |
| 10049 | WSAEADDRNOTAVAI | アドレスが無効です |
| 10050 | WSAENETDOWN | ネットワークがダウンしています |
| 10051 | WSAENETUNREACH | ネットワークに到達できません |
| 10052 | WSAENETRESET | ネットワーク接続が破棄されました |
| 10053 | WSAECONNABORTED | ソフトウェアによって切断されました |
| 10054 | WSAECONNRESET | 接続先で切断されました |
| 10055 | WSAENOBUFS | バッファが取れません |
| 10056 | WSAEISCONN | 既に接続されています |
| 10057 | WSAENOTCONN | ソケットが接続されていません |
| 10058 | WSAESHUTDOWN | ソケットシャットダウン後に送信しています |
| 10060 | WSAETIMEDOUT | 接続がタイムアウトされました |
| 10061 | WSAECONNREFUSED | 接続拒否されました |
| 10064 | WSAEHOSTDOWN | ホストがダウンしています |
| 10065 | WSAEHOSTUNREACH | ホストまで到達できません |
| 10067 | WSAEPROCLIM | プロセスが多すぎます |
| 10091 | WSASYSNOTREADY | ネットワークサブシステムが使用できません |
| 10092 | WSAVERNOTSUPPORTED | サポートされていないバージョンです |
| 10093 | WSANOTINITIALISED | WSAStartupは成功したが実行されていません |
| 10101 | WSAEDISCON | 終了処理中 |
| 11001 | WSAHOST_NOT_FOUND | ホストが見つかりません |
| 11002 | WSATRY_AGAIN | サーバーの異常です |
| 11003 | WSANO_RECOVERY | 復帰できないエラーです |

# 第３編

# ホストE-LINKサンプルライブラリ

# アルファベット順ライブラリ目次


| 項番 | ライブラリ名 | ページ |
|---|---|---|
| 1 | ELINK_checkccmdack | 38 |
| 2 | ELINK_cload | 36 |
| 3 | ELINK_cupload | 37 |
| 4 | ELINK_gets | 34 |
| 5 | ELINK_headercheck | 39 |
| 6 | ELINK_outccmd | 35 |
| 7 | ELINK_puts | 33 |

## ■ホストE-LINKサンプルライブラリ

### E-LINK送信

```
int ELINK_puts(SOCKET socket_n, char *s, int buffersize, short time );
```

### E-LINK受信

```
int ELINK_gets(SOCKET socket_n, char *s, int buffersize, short time);
```

### システムコマンド発行

```
int ELINK_outccmd(SOCKET socket_n, char *s, short time);
```

### ファイルの送信

```
int ELINK_cload(SOCKET socket_n, FILE *fp, void (*func)(long totalsize,
                                                        long size), short time);
```

### ファイルの受信

```
int ELINK_cupload(SOCKET socket_n, FILE *fp, void (*func)(long totalsize,
                                                        long size) ,short time);
```

### コマンド応答チェック

```
int ELINK_checkccmdack(SOCKET socket_n, short time);
```

### ファイル受信時のヘッダチェック

```
int ELINK_headercheck(SOCKET socket_n, short time);
```

Terminal_Sample.c は、Winsock ライブラリを使用して作成したコンソールプログラム例です。
Windows95、Watcom 11.0J で確認済みです。

Terminal_Sample.c は、ホスト E-LINK サンプル、ターミナルＦＤに含まれています。

Ｅ－ＬＩＮＫのホストライブラリはプラットホームの依存性を少なくするために、ライブラリ提供はせずにソースコードを全て提供し、**サンプルライブラリ**と言う形で提供いたします。

以下は Winsock で作成した９０４システムをＥ－ＬＩＮＫでコントロールするサンプルライブラリの説明です。

# *ELINK_puts*

## 機　能

E-LINK 送信

## 形　式

int ELINK_puts(SOCKET ***socket_n***, char ****s***, int ***buffersize***, short ***time*** );

## 解　説

① ***socket_n*** はソケット番号です。

② ****s*** は送信するバッファの先頭アドレスです。

③ ***buffersize*** は送信するバッファのサイズです。

④ ***time*** はタイムアウトする時間（秒）です。

## 戻り値

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| １ | TRUE | 正常終了 |
| －１ | SOCKET_ERROR | WSAGetLastErrorで確認してください |
| －１０ | ELINK_TIME_OUT | タイムアウト |

## 留意事項

○全てのデータを送信するまでブロッキングされます。

○time が０以下の場合、送信し終えるまでタイムアウトエラーはありません。

# *ELINK_gets*

## 機　能

E-LINK 受信

## 形　式

int ELINK_gets(SOCKET ***socket_n***, char ****s***, int ***buffersize***, short ***time***);

## 解　説

① ***socket_n*** はソケット番号です。

② ****s*** は送信するバッファの先頭アドレスです。

③ ***buffersize*** は送信するバッファのサイズです。

④ ***time*** はタイムアウトする時間（秒）です。

## 戻り値

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| １～ | ありません | 受信サイズ |
| ０ | ありません | 切断された |
| －１ | SOCKET_ERROR | WSAGetLastErrorで確認してください |
| －１０ | ELINK_TIME_OUT | タイムアウト |

## 留意事項

○time が負値の場合は、データを受信するまでブロッキングします。

# *ELINK_outccmd*

**機　能**　　システムコマンド発行

**形　式**　　int ELINK_outccmd(SOCKET ***socket_n***, char ****s***, short ***time***);

**解　説**

① ***socket_n*** はソケット番号です。

② ****s*** は送信するコマンドの先頭アドレスです。

③ ***time*** はタイムアウトする時間（秒）です。

**戻り値**

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| １ | TRUE | 正常終了 |
| －１ | SOCKET_ERROR | WSAGetLastErrorで確認してください |
| －９ | MEMORY_ALLOC_ERROR | メモリ不足 |
| －１０ | ELINK_TIME_OUT | タイムアウト |

**留意事項**　○ターミネートは内部で付加します。

# *ELINK_cload*

## 機　能

ファイルの送信

## 形　式

```
int ELINK_cload(SOCKET socket_n, FILE *fp, void (*func)(long totalsize,
                                                long size), short time);
```

## 解　説

① ***socket_n*** はソケット番号です。

② ****fp*** はファイルディスクリプタです。(rb でオープンしてください)

③ ****func*** はユーザ定義の関数のポインタです。
送信中に処理したいこと、送信の進み具合を示すインジケーターなどを行う
関数への引き数として、

**totalsize**
送信データの総数（最初の呼出時のみ有効、以降は０）

**size**
今回送信した数を渡します。

この関数は送信の合間に呼び出されます。
（関数のポインタとしてＮＵＬＬが指定されると呼び出しません。）

④ ***time*** はタイムアウトする時間（秒）です。

## 戻り値

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| １ | TRUE | 正常終了 |
| ０ | ありません | 切断された |
| －１ | SOCKET_ERROR | WSAGetLastErrorで確認してください |
| －８ | FILE_HEADER_ERROR | ファイルヘッダエラー |
| －１０ | ELINK_TIME_OUT | タイムアウト |

## 留意事項

ありません。

# *ELINK_cupload*

## 機　能

ファイルの受信

## 形　式

```
int ELINK_cupload(SOCKET socket_n, FILE *fp, void (*func)(long totalsize,
                                              long size) ,short time);
```

## 解　説

① ***socket_n*** はソケット番号です。

② ****fp*** はファイルディスクリプタです。(wb でオープンしてください)

③ ****func*** はユーザ定義の関数のポインタです。
受信中に処理したいこと、受信の進み具合を示すインジケーターなどを行う
関数への引き数として、

**totalsize**
受信データの総数（最初の呼出時のみ有効、以降は０）

**size**
今回受信した数を渡します。

この関数は受信の合間に呼び出されます。
（関数のポインタとしてＮＵＬＬが指定されると呼び出しません。）

④ ***time*** はタイムアウトする時間（秒）です。

## 戻り値

| 値 | 定　　数 | 意　　味 |
|---|---|---|
| 1 | TRUE | 正常終了 |
| ０ | ありません | 切断された |
| －１ | SOCKET_ERROR | WSAGetLastErrorで確認してください |
| －８ | FILE_HEADER_ERROR | ファイルではない |
| －１０ | ELINK_TIME_OUT | タイムアウト |

## 留意事項

ありません。

# *ELINK_checkccmdack*

## 機　能

コマンド応答チェック

## 形　式

int ELINK_checkccmdack(SOCKET ***socket_n***, short ***time***);

## 解　説

① ***socket_n*** はソケット番号です。

② ***time*** はタイムアウトする時間（秒）です。

## 戻り値

| 値 | 定　数 | 意　味 |
|---|---|---|
| '!' | ありません | コマンド正常 |
| '%' | ありません | コマンド異常 |
| '?' | ありません | 未定義コマンド |
| ０ | ありません | 切断された |
| －１ | SOCKET_ERROR | WSAGetLastErrorで確認してください |
| －１０ | ELINK_TIME_OUT | タイムアウト |

## 留意事項

ありません。

# *ELINK_headercheck*

## 機　能

ファイル受信時のヘッダチェック

## 形　式

int ELINK_headercheck(SOCKET ***socket_n***, short ***time***);

## 解　説

① ***socket_n*** はソケット番号です。

② ***time*** はタイムアウトする時間（秒）です。

## 戻り値

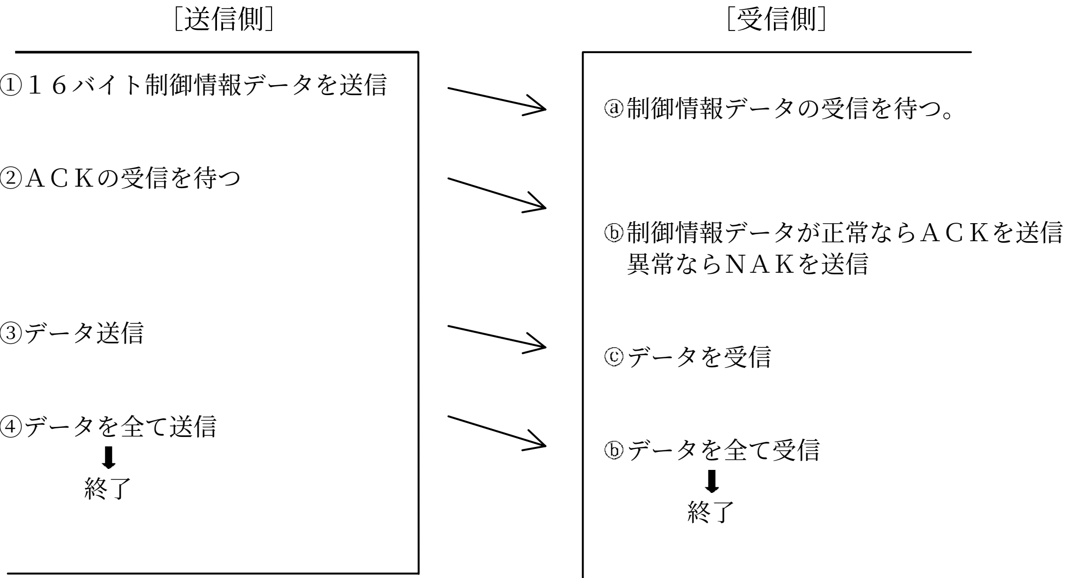
| 値 | 定　　数 | 意　　味 |
|---|---|---|
| 0x06 | ありません | ACK |
| 0x15 | ありません | NAK |
| ０ | ありません | 切断された |
| －１ | SOCKET_ERROR | WSAGetLastErrorで確認してください |
| －１０ | ELINK_TIME_OUT | タイムアウト |

## 留意事項

ありません。

# ファイル転送方法

## Ｅ－ＬＩＮＫの場合のファイルデータ送受信手順




［注１］再送の回数は最大１０回とする。
送信側が再送を１０回繰り返しても送信ステータスが「送信完」にならない場合は異常終了として強制中断する。また、ＲＥＴＲＹ可能でなければ異常終了として強制中断する。

［注２］タイムアウトは３０秒とする。
受信側で、受信待ち時間が３０秒を越えた時点で異常終了として強制中断する。
送信側で、送信ステータスの「送信中」が３０秒を越えた時点で異常終了として強制中断する。

## Ｅ－ＬＩＮＫの制御情報データ構造

| オフセット | 内容 |
|---|---|
| ＋０ | |
| １ | ロードアドレス |
| ２ | （プログラムファイルの転送時のみ有効） |
| ３ | |
| ４ | |
| ５ | ファイルサイズ |
| ６ | |
| ７ | |
| ８ | |
| ９ | Ｓｙｓｔｅｍ　ｕｓｅ |
| １０ | |
| １１ | |
| １２ | ファイル属性 |
| １３ | |
| １４ | Ｓｙｓｔｅｍ　ｕｓｅ |
| １５ | |

# 通信速度

Ｅ－ＬＩＮＫの通信速度は、以下のようになります。

◎100Kbyte のファイルを送信したとき約５秒
この数値はネットワークの負荷にもよります。

**ホスト*E-LINK* 説明書**

１９９９年１１月第１版第１刷発行

**発行所　　株式会社ファースト**

本　　社　〒242-0001　神奈川県大和市下鶴間２７９１－５

テクニカルサポート　**FAX　046-272-8692**　TEL　046-272-8691

**E-mail : support@fast-corp.co.jp**

B-000845